Large-scale Processing of Streaming Data

Qingsong Guo

May 10, 2018 SCST, North University of China

Education Background

B.S Sep 2003 – Jul 2007

- North University of China
- Department of Computer Science

M.S Sep 2003 – Jul 2007

- Renmin University of China
- Prof. Xiaofeng Meng
- Lab of Web And Mobile Data Management(WAMDM), Info School

Ph.D   2011.9 – 2016.8

- University of Southern Denmark
- Prof. Yongluan Zhou
- Department of Mathematics and Computer Science, Faculty of Science

My Research

My research can be subsumed under Big Data

Semi-structured data management

- Index, query optimization, keyword search
- Implementation of native XML database “OrientX”

Large-scale Processing of Streaming Data

- Massive parallelization,
- Resource optimization, operator placement
- Stateful load balancing

Interactive Analysis of Big Data

- Approximate Query Processing(AQP)
- Multiscale approximation & analysis
- Multiscale dissemination of streaming data

Big Graph Analytics

- Temporal Graph Analysis

Outline

1	**Why Big Data?**
2	**Big Data Fundamentals**
3	**Big Streaming Computation**
4	**Conclusion**

1

Why Big Data?

Backgrounds For Big Data

Data Management & Data Analysis

Observation(观察) → Data (数据) → Data analysis (数据分析)

Kepler's Laws of Planetary Motion

开普勒行星三定律

Beers and Diapers

啤酒和尿布

AlphaGo Deep Learning

人机对弈和深度学习

History of Data Management

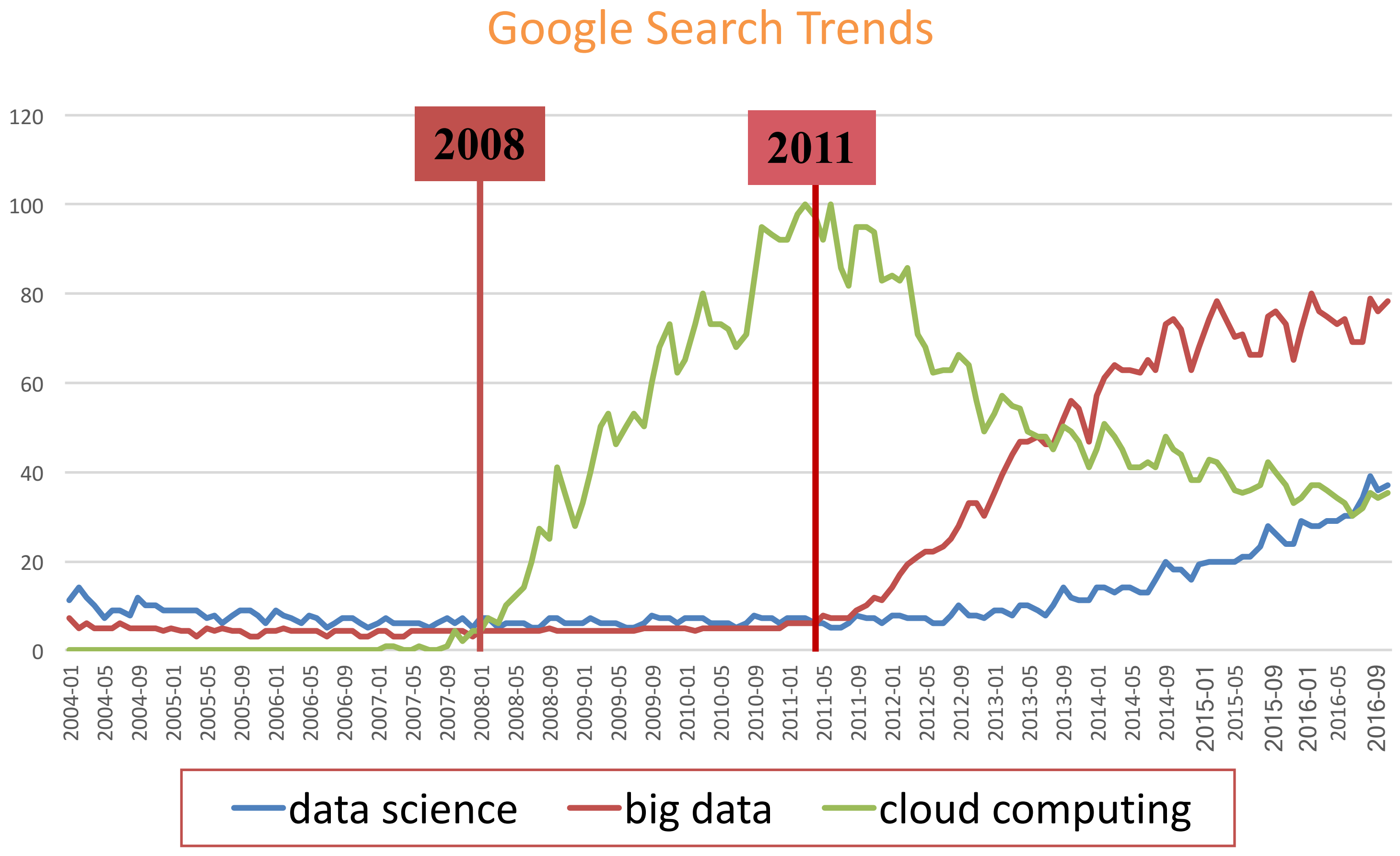
Prehistory
- Invention of digital computer
- 1900-1970's

Database
- 1971, E.F. Codd proposed the “Relation Model”
- Data schema, view, logical independency, physical independency

Cloud Computing
- 2005, Google
- MapReduce, Large-scale cluster computing
- IaaS, PaaS, SaaS
- NoSQL

Big Data & Data Science
- 2011
- Batch processing, interactive analysis, streaming processing
- Statistical Inference, Data Mining, Machine Learning

The Search Trends

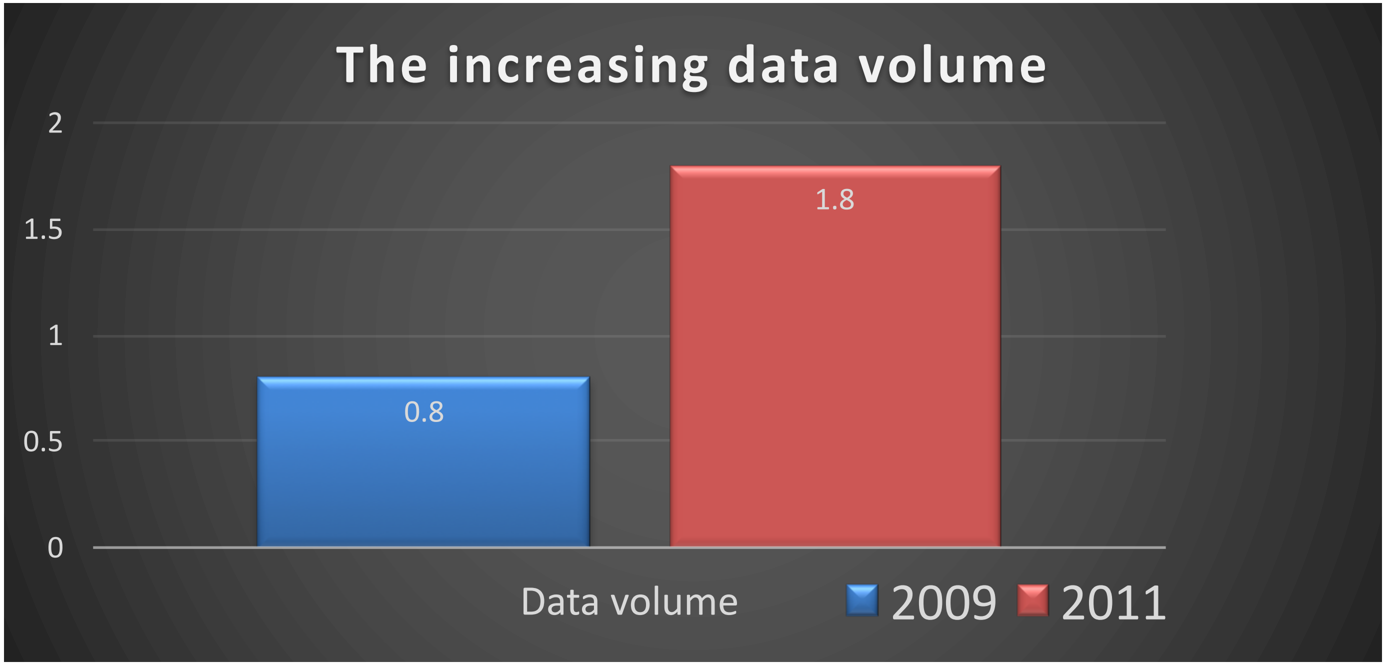


The Rise of Big Data

Data volume(IDC's report)

– 800,000 PB in 2009
– 1.8 zettabytes (1.8 million petabytes) in 2011
– 50 fold by 2020

1 PB = 1000TB
1 TB = 1000GB
1 GB = 1000MB



Big Data Examples

1. Scientific data

Scientific Equipment	Data Rate
2.5m Telescope	200 GB/day
LHC(Large Hadron Collider)	300 GB/sec
Astrophysics Data	10 PB/year
Ion Mobility Spectroscopy	10 TB/day
3D X-ray Diffraction Microscopy	24 TB/day
GPS(Personal Location Data)	1 PB/year

2. Web & Social Network Data

What is big data used for?

Reports, e.g.,

– Track business processes, transactions

Diagnosis, e.g.,

– Why is user engagement dropping?

– Why is the system slow?

– Detect spam, worms, viruses, DDoS attacks

Decisions, e.g.,

– Personalized medical treatment

– Decide what feature to add to a product

– Decide what ads to show

Data is only as useful as the decisions it enables

– 中国移动只能查询最近三个月的消费记录

– 1950s美国为了保存和查询用户信息发明数据库

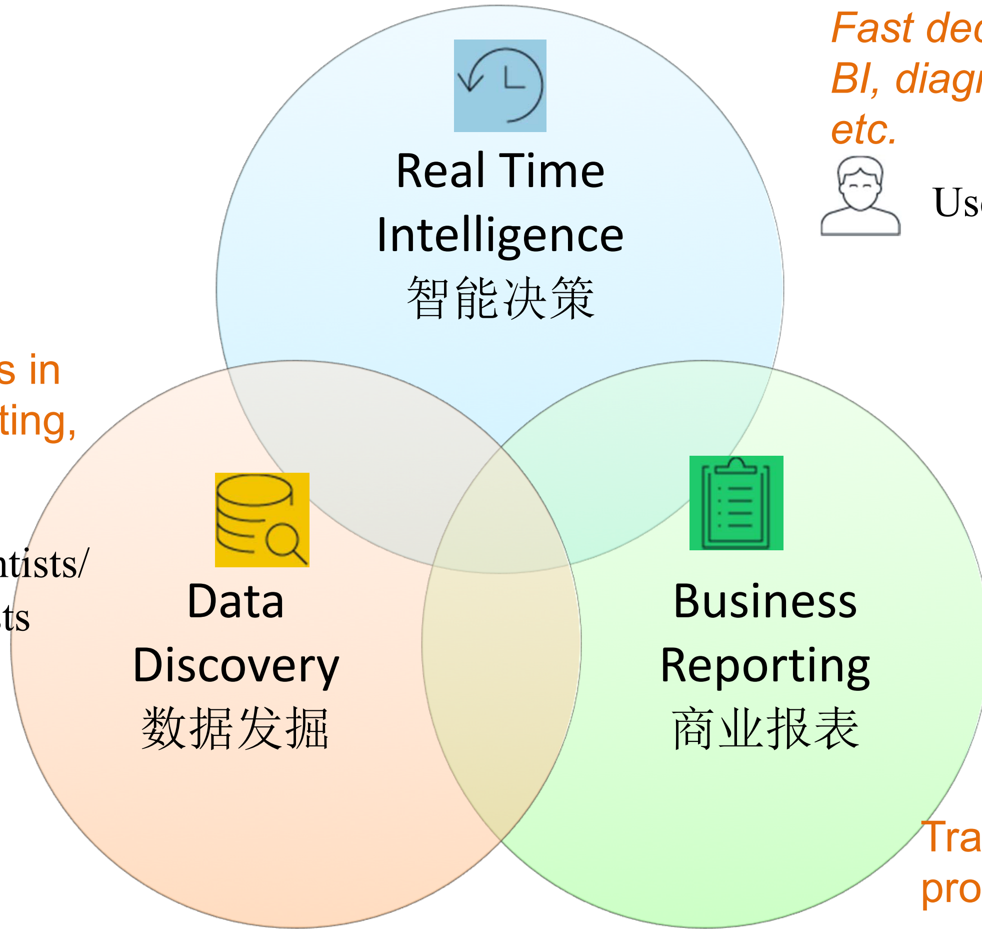
What is Big Data Used for?

Fast decision-making in BI, diagnosis in security, etc.

Users

In-depth analysis in scientific computing, etc.

Data Scientists/
Analysts



Track business processes, transactions

Data is only as useful as the decisions it enables

Business Users

The Story of Google

Larry Page and Sergey Brin created Google in 1998

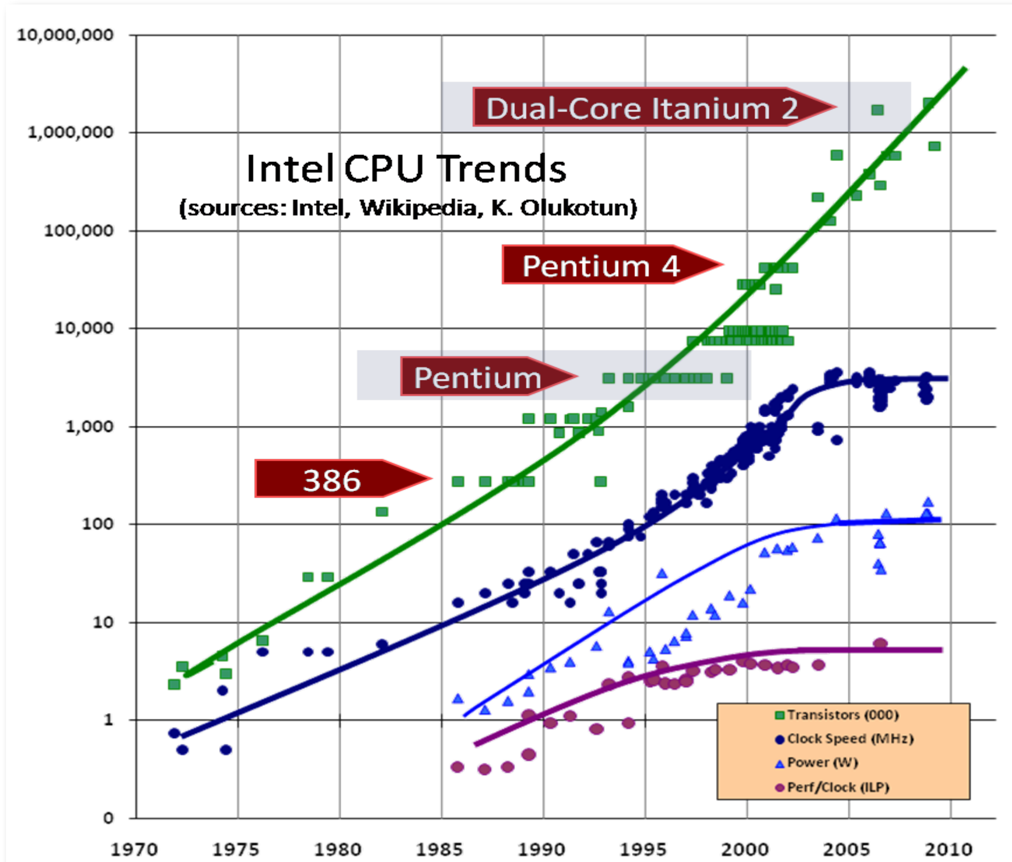
- Over 1 billion webpages
- Classmate Sean Anderson proposed "Googol"
- Larry mis-registered "Googol" as "Google"

What "Googol" stands for?

- Astronomical number of 1 followed by 100 zeros (10^{100})
- In 1938, an American mathematician Edwards Kasner was wandering a name for that number, and his nephew coined that odd term "googol"

The Free Lunch Is Over – Moore's Law Fails

Chairman of ISO C++ Standard Committee
"C++ Coding Standards"
"Exceptional C++"
"More Exceptional C++"
"Exceptional C++ Style"



Intel CPU Introductions

Herb Sutter. The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software. March 2005.

Data-Intensive System Challenge

For computation that accesses 1 TB in 5 minutes

- Data distributed over 100+ disks
  - Assuming uniform data partitioning
- Compute using 100+ processors
- Connected by gigabit Ethernet (or equivalent)

System requirements

- Lots of disks
- Lots of processors
- Low-latency network delay
  - fast, local-area network access

High Performance Computing

High performance computing (HPC)

- High Performance Computer: Supercomputer TOP500 List
- Quantum Computing

Rank	Cores	Max, Peak (PFlop/s)	Name	Country
1	10,649,600	93.015, 125.436	TaihuLight	China
2	3,120,000	33.863, 54.902	Tianhe-2	China
3	361,760	19.590, 25.326	Piz Daint	Switzerland
4	19,860,000	19.135, 28.129	Gyoukou	Japan
5	560,640	17.590, 27.113	Titan	US
...		...	...	...

Cluster Computing

- High Performance Supercomputer is expensive
  - The world just need 3 super-computer, Thomas Watson, IBM CEO
  - 256KB is enough in year 2000, Bill Gates
- Cluster is consist of many commodity machine
  - Failure for commodity computers is inevitable

	Notebooks		PCs	
Year	2005-2006	2003-2004	2005-2006	2003-2004
1	5	7	15	20
4	12	15	22	28

Annual Failure Rates of PCs, Gartner Dataquest (June 2006)

Question: Suppose we have a cluster of 2,000 commodity machines, how many machines would failed per day in 2005?

Why Big Data Now?

1. Low cost storage to store data that was discarded earlier
2. Powerful multi-core processors (commodity computer)
3. Low latency possible by distributed computing: Compute clusters and grids connected via high-speed networks
4. Virtualization → Partition, Aggregate, isolate resources in any size and dynamically change it → Minimize latency for scaling
5. Affordable storage and computing with minimal man power via clouds → Possible because of advances in Networking

Why Big Data Now? (Cont.)

6. Better understanding of task distribution (MapReduce), computing architecture (Hadoop),
7. Advanced analytical techniques (Machine learning)
8. Managed Big Data Platforms
   – *Cloud service providers, such as AWS provide Elastic MapReduce, Simple Storage Service (S3) and HBase – column oriented database. Google BigQuery and Prediction API.*
9. Open-source software: OpenStack, PostGreSQL
10. Support from government: March 12, 2012: Obama announced $200M for Big Data research. Distributed via NSF, NIH, DOE, DoD, DARPA, and USGS (Geological Survey)

How Much do You Know?

Cloud Computing?

MapReduce, GFS, Bigtable, Chubby

Hadoop, Zookeeper, Hive, Pig

S3, Dynamo, Amazon Web Services (AWS)

Yarn, Mesos, …

Big Data?

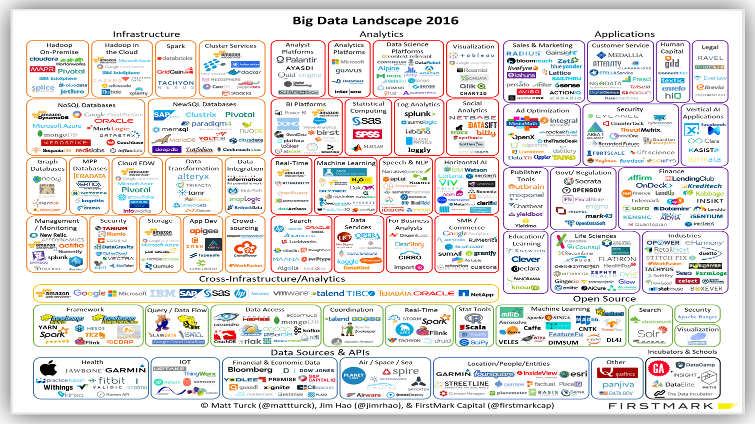
Spark, Spark Streaming

Apache Storm, Smaza,

Flink, SummingBird, Google's Dataflow

GraphX, GraphLab

…



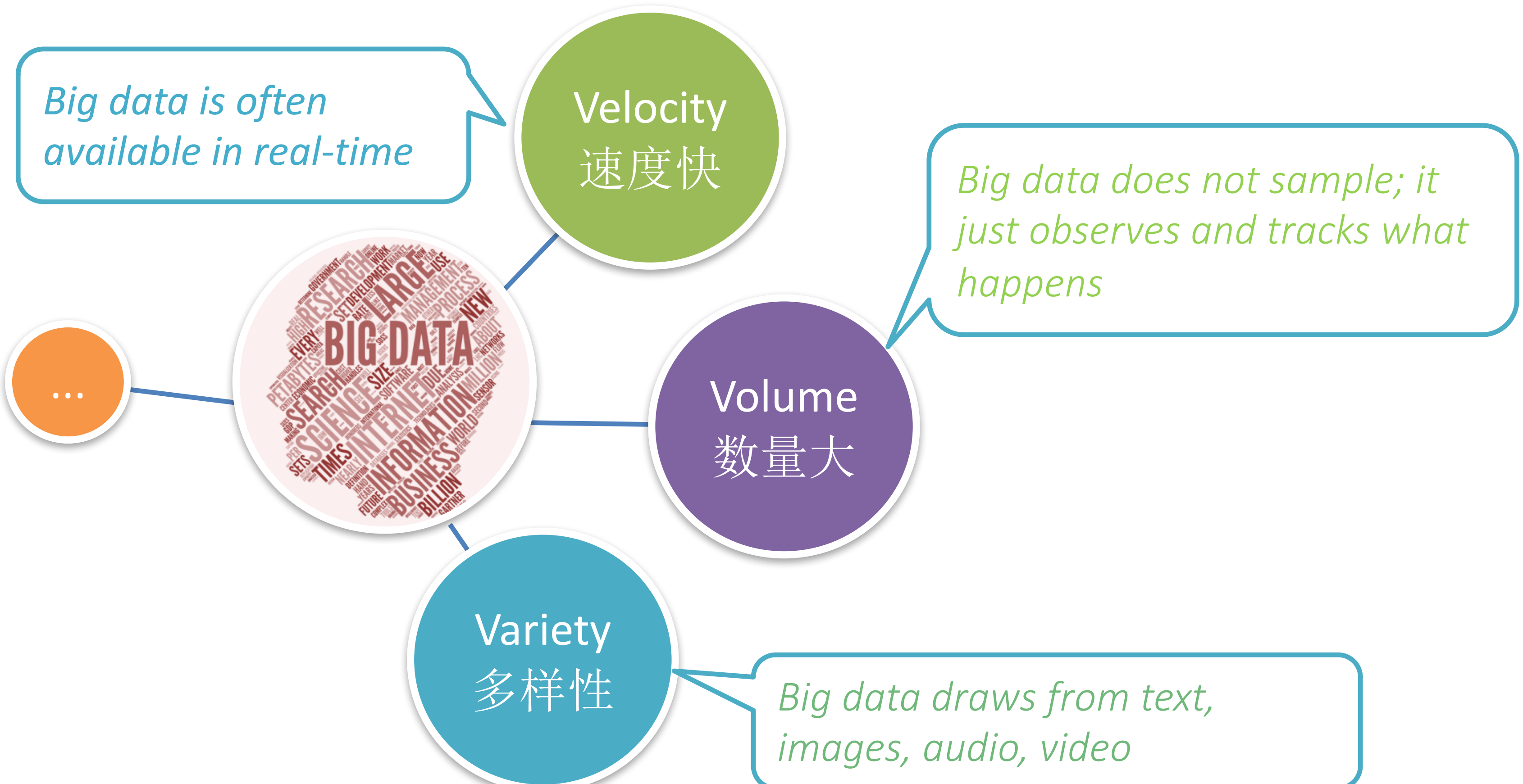

Menu
amazon web services
AWS re:Invent
Products
Solutions
Pricing
Software
More
English
My Account
Create an AWS Account
Compute
Amazon EC2
Amazon EC2 Container Registry
Amazon EC2 Container Service
Amazon Lightsail
Amazon VPC
AWS Batch
AWS Elastic Beanstalk
AWS Lambda
Auto Scaling
Elastic Load Balancing
Storage
Amazon Simple Storage Service (S3)
Amazon Elastic Block Storage (EBS)
Amazon Elastic File System (EFS)
Amazon Glacier
AWS Storage Gateway
AWS Snowball
AWS Snowball Edge
AWS Snowmobile
Database
Amazon Aurora
Amazon RDS
Amazon DynamoDB
Amazon ElastiCache
Migration
AWS Database Migration Service
AWS Server Migration Service
AWS Snowball
AWS Snowball Edge
AWS Snowmobile
Networking & Content Delivery
Amazon VPC
Amazon CloudFront
Amazon Route 53
AWS Direct Connect
Elastic Load Balancing
Developer Tools
AWS CodeCommit
AWS CodeBuild
AWS CodeDeploy
AWS CodePipeline
AWS X-Ray
AWS Command Line Interface
Management Tools
Amazon CloudWatch
Amazon EC2 Systems Manager
AWS CloudFormation
AWS CloudTrail
Security & Identity, Compliance
AWS Identity and Access Management (IAM)
Amazon Inspector
AWS Certificate Manager
AWS CloudHSM
AWS Directory Service
AWS Key Management Service
AWS Organizations
AWS Shield
AWS WAF
Analytics
Amazon Athena
Amazon EMR
Amazon CloudSearch
Amazon Elasticsearch Service
Amazon Kinesis
Amazon Redshift
Amazon QuickSight
AWS Data Pipeline
AWS Glue
Artificial Intelligence
Amazon Lex
Amazon Polly
Amazon Rekognition
Amazon Machine Learning
Application Services
AWS Step Functions
Amazon API Gateway
Amazon Elastic Transcoder
Amazon AppStream
Messaging
Amazon SQS
Amazon Pinpoint
Amazon SES
Amazon SNS
Business Productivity
Amazon WorkDocs
Amazon WorkMail
Desktop & App Streaming
Amazon WorkSpaces
Amazon AppStream 2.0
Software
AWS Marketplace
Internet of Things
AWS IoT Platform
AWS Greengrass
AWS IoT Button
NEW PROD
AWS re:Inve
amazon web services
Azure
Google Cloud Platform
阿里云
HOT
最新活动
产品
解决方案
云市场
大数据
社区
支持
更多
控制台
备案
登录
云计算基础服务
大数据（数加） >
安全（云盾） >
域名与网站（万网） >
弹性计算
数据库 >
存储与CDN
分析
云通信
网络
管理与监控
应用服务
互联网中间件
移动服务
视频服务
云服务器 ECS
可弹性扩展、安全、稳定、易用的计算服务
块存储
可弹性扩展、高性能、高可靠的块级随机存储
专有网络 VPC
帮您轻松构建逻辑隔离的专有网络
负载均衡
对多台云服务器进行流量分发的负载均衡服务
弹性伸缩
自动调整弹性计算资源的管理服务
资源编排
批量创建、管理、配置云计算资源
容器服务
应用全生命周期管理的 Docker 服务
高性能计算 HPC
加速深度学习、渲染和科学计算的 GPU 物理机
批量计算
简单易用的大规模并行批处理计算服务
阿里云
DaoCloud

2 Big Data Fundamentals

Terminology, Key Technologies

Essentials of Big Data

- 3Vs, 4Vs, 5Vs:
  - *Volume: TB, PB, EB, …*
  - *Velocity: TB/sec. Speed of creation or change*
  - *Variety: Type (Text, audio, video, images, geospatial, …)*



Challenges for Big Data Analytics

1. Affordable Price (廉价性)

Commodity cluster vs High performance computer (HPC)

Pay-As-You-Go pricing model

2. Fault tolerance (容错)

How could a cluster of computers coordinate with each other to handle a big data problem?

3. Scalability (可扩展性)

How is an application scales out to thousands computers?

4. Elasticity (弹性计算)

Elastic management of computing resources

Adaptive scale-out/scale-in, scale-up/down

Cloud Services

Applications	**Software as a service (SaaS)** 软件即服务 Operating environment largely is a software delivery methodology that provides licensed multi-tenant access to software and its functions remotely as a Web-based service.
Frameworks	**Platform as a service (PaaS)** 平台即服务 Provides all of the facilities required to support the complete life cycle of building and delivering web applications and services entirely from the Internet.
Hardware	**Infrastructure as a service (IaaS)** 基础架构即服务 Delivery of technology infrastructure as an on demand scalable service.

ACID Requirements

Atomicity:

- *All or nothing. If anything fails, entire transaction fails. Example, Payment and ticketing.*

Consistency

- *If there is error in input, the output will not be written to the database. Database goes from one valid state to another valid states. Valid=Does not violate any defined rules.*

Isolation

- *Multiple parallel transactions will not interfere with each other.*

Durability

- *After the output is written to the database, it stays there forever even after power loss, crashes, or errors.*

Relational databases provide ACID while non-relational databases aim for BASE (Basically Available, Soft, and Eventual Consistency)

Types of Data

- Structured Data
  - Data that has a pre-set format, e.g., Address Books, product catalogs, banking transactions,
- Semi-Structured Data & Unstructured Data
  - Data that has no pre-set format. Movies, Audio, text files, web pages, computer programs, social media,
  - Unstructured data that can be put into a structure by available format descriptions
  - 80% of data is unstructured.
- Metadata: Definitions, mappings, scheme of data
- Batch vs. Streaming Data
  - Real-Time Data: Streaming data that needs to analyzed as it comes in. E.g., Intrusion detection. Aka "Data in Motion"
  - Data at Rest: Non-real time. E.g., Sales analysis.

Ref: Michael Minelli, "Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today's Businesses," Wiley, 2013, ISBN:'111814760X

Relational Databases and SQL

Relational Database

- Stores data in tables. A "Schema" defines the tables, the fields in tables and relationships between the two. Data is stored one column/attribute

Order tables

Order Number	Customer ID	Product ID	Quantity	Unit Price
…	…	…	…	…

Customer tables

Customer ID	Customer Name	Customer Address	Gender	Income Range
…	…	…	…	…

SQL (Structured Query Language):

- Most commonly used language for creating, retrieving, updating, and deleting (CRUD) data in a relational database

Example: To find the gender of customers who bought XYZ

Select CustomerID, State, Gender, ProductID

from "Customer Table", "Order Table"

where ProductID = XYZ

Non-relational Databases

NoSQL: Not Only SQL

- Database that uses non-SQL interfaces, e.g., Python, etc. for retrieval.
- Typically store data in key-value pairs.
- Not limited to rows or columns. Data structure and query is specific to the data type
- RESTful (Representational State Transfer) web-like APIs
- Eventual consistency: BASE in place of ACID

NewSQL Database

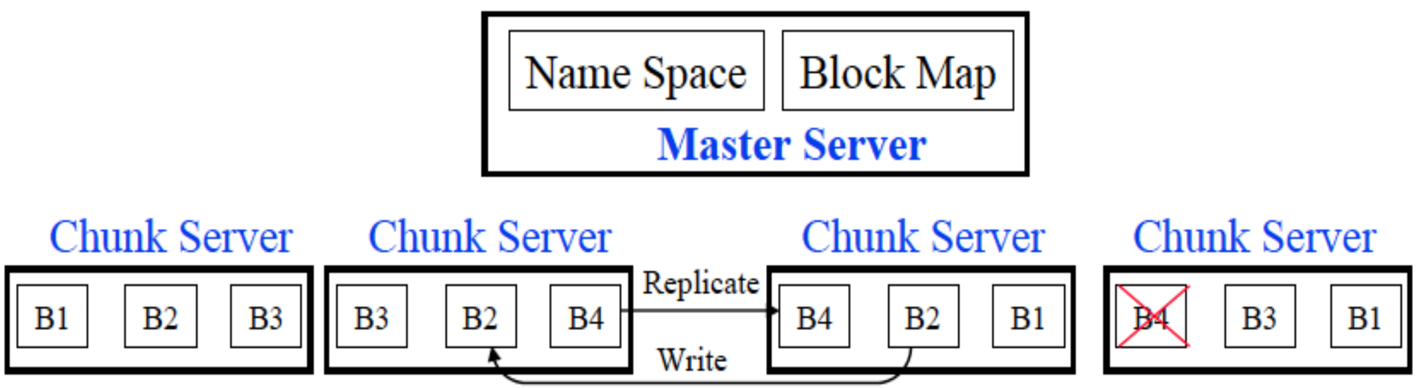
- Overcome scaling limits of Relational Database
- Same scalable performance as NoSQL but using SQL
- Providing ACID
- Also called Scale-out SQL
- Generally use distributed processing.

Types of Databases

- Relational Databases: PostgreSQL, SQLite, MySQL
- NewSQL Databases: Scale-out using distributed processing
- Non-relational Databases:
  - Key-Value Pair (KVP) Databases: Data is stored as Key:Value, e.g., Riak Key-Value Database
  - Document Databases: Store documents or web pages, e.g.,MongoDB, CouchDB
  - Columnar Databases: Store data in columns, e.g., HBase
  - Graph Databases: Stores nodes and relationship, e.g., Neo4J
  - Spatial Databases: For map and nevigational data, e.g., OpenGEO, PortGIS, ArcSDE
  - In-Memory Database: All data in memory. For real time applications
  - Cloud Databases: Any data that is run in a cloud using IAAS, VM Image, DAAS

Google File System

- GFS is a Distributed File System
- Commodity computers serve as "Chunk Servers" and store multiple copies of data blocks
- A master server keeps a map of all chunks of files and location of those chunks.
- All writes are propagated by the writing chunk server to other chunk servers that have copies.
- Master server controls all read-write accesses



Ref: S. Ghemawat, et al., "The Google File System", OSP 2003, http://research.google.com/archive/gfs.html

BigTable

- GFS provides a distributed storage system
- Data stored in rows and columns
- Optimized for sparse, persistent, multidimensional sorted map.
- Uses commodity servers
- Not distributed outside of Google but accessible via Google App Engine

Ref: F. Chang, et al., "Bigtable: A Distributed Storage System for Structured Data," 2006, http://research.google.com/archive/bigtable.html

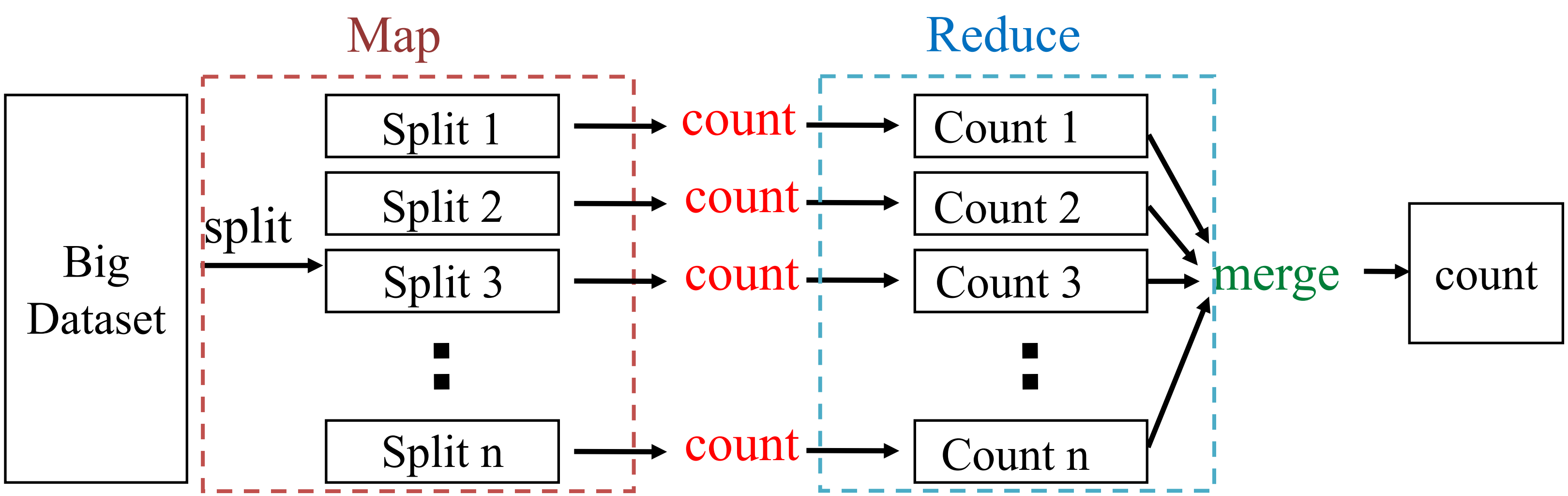
MapReduce in a Nutshell

- Programming model for processing massive amounts of data in parallel
  - Simple but effective
- Design Goals:
  - Distributed: over a large number of inexpensive processors
  - Scalable: can expand or contract as needed, so as to exploit a large set of commodity machines, hundreds or even thousands
  - Fault tolerant: Continue in spite of some failures to offer high availability

Ref: J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," OSDI 2004, http://research.google.com/archive/mapreduce-osdi04.pdf

MapReduce in a Nutshell (Cont.)

- Map(): Takes a set of data and converts it into another set of key-value pairs.
- Reduce(): Takes the output from Map as input and outputs a smaller set of key-value pairs.



MapReduce Example 1

Dataset at hand

- 100 files with daily temperature in two cities. Each file has 10,000 entries. For example, one file may have (Toronto 20), (New York 30),…

Task to complete

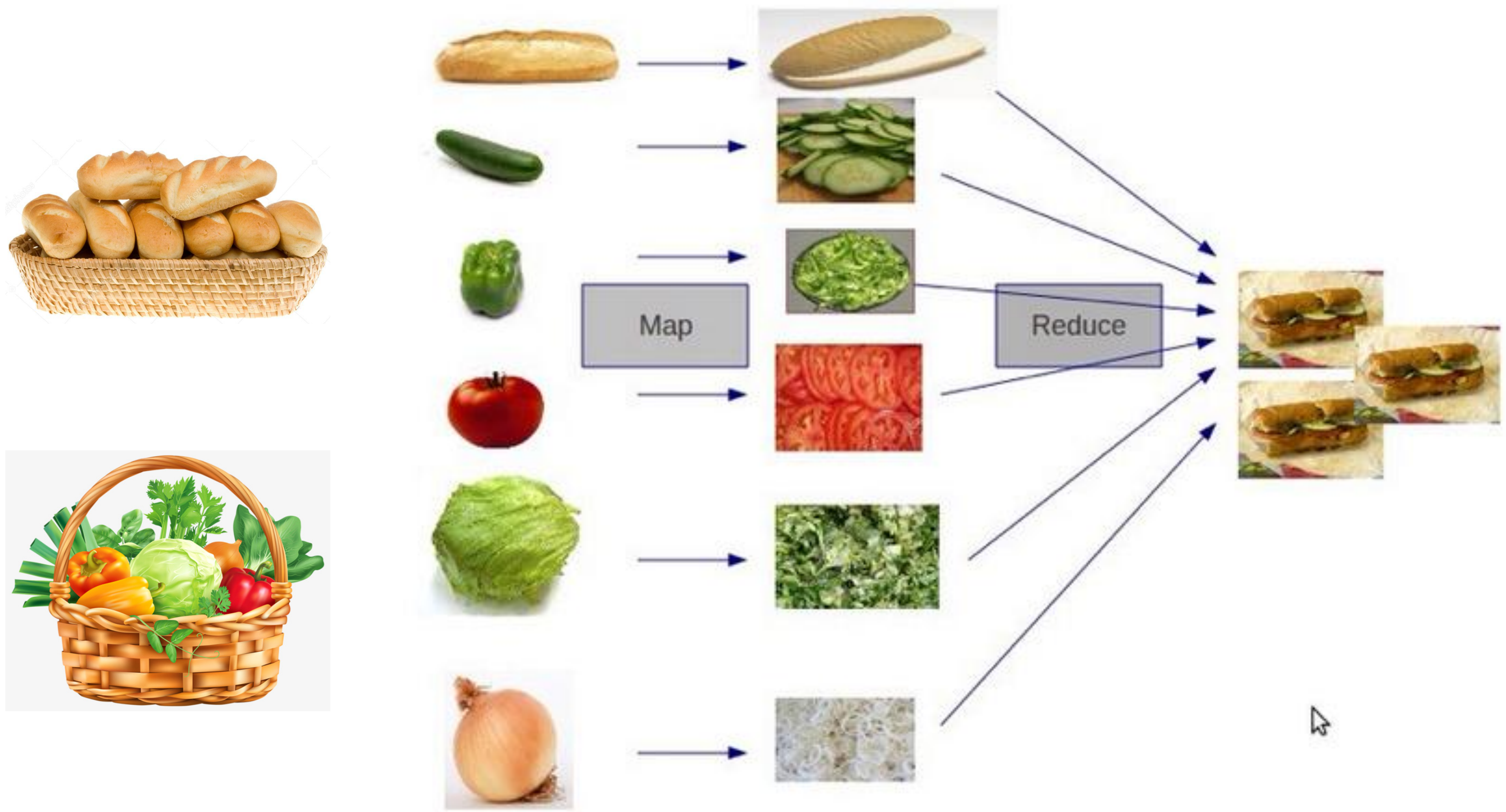
- to compute the maximum temperature in the two cities.

Algorithm

- Assign the task to 100 Map processors each works on one file. Each processor outputs a list of key-value pairs, e.g., <Toronto, 30>, <New York, 65>, …
- Now we have 100 lists each with two elements. We give this list to two reducers – one for Toronto and another for New York.
- The reducer produce the final answer: <Toronto, 55>, <New York 65>

Ref: IBM. "What is MapReduce?." http://www-01.ibm.com/software/data/infosphere/hadoop/mapreduce/

Example 2: Making Sandwich



MapReduce Optimization

Scheduling

- Task is broken into pieces that can be computed in parallel
- Map tasks are scheduled before the reduce tasks.
- If there are more map tasks than processors, map tasks continue until all of them are complete.
- A new strategy is used to assign Reduce jobs so that it can be done in parallel The results are combined.

Synchronization

- The map jobs should be comparable so that they finish together. Similarly reduce jobs should be comparable.

Code/Data Collocation

- The data for map jobs should be at the processors that are going to map.

Fault/Error Handling

- If a processor fails, its task needs to be assigned to another processor.

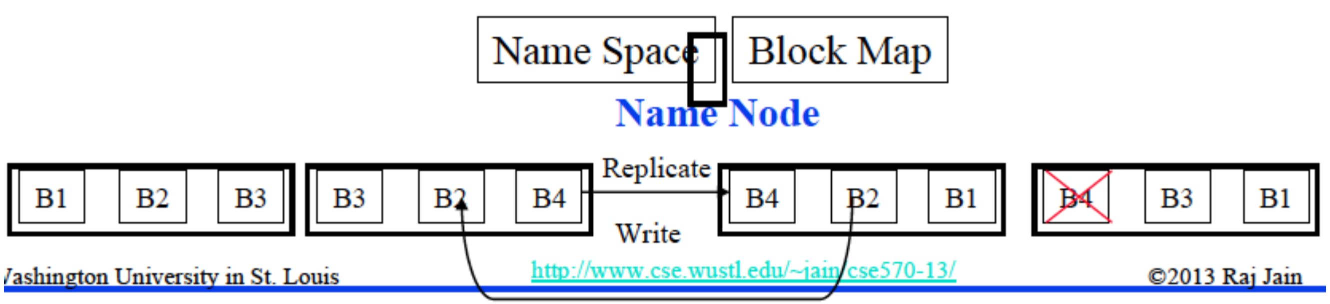
Story of Hadoop

- Doug Cutting at Yahoo and Mike Caferella were working on creating a project called “Nutch” for large web index.
- They saw Google papers on MapReduce and Google File System and used it
- Hadoop was the name of a yellow plus elephant toy that Doug’s son had.
- In 2008 Amr left Yahoo to found Cloudera.
- In 2009 Doug joined Cloudera.

Ref: Michael Minelli, “Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today‘s Businesses,” Wiley, 2013, ISBN:'111814760X

Hadoop

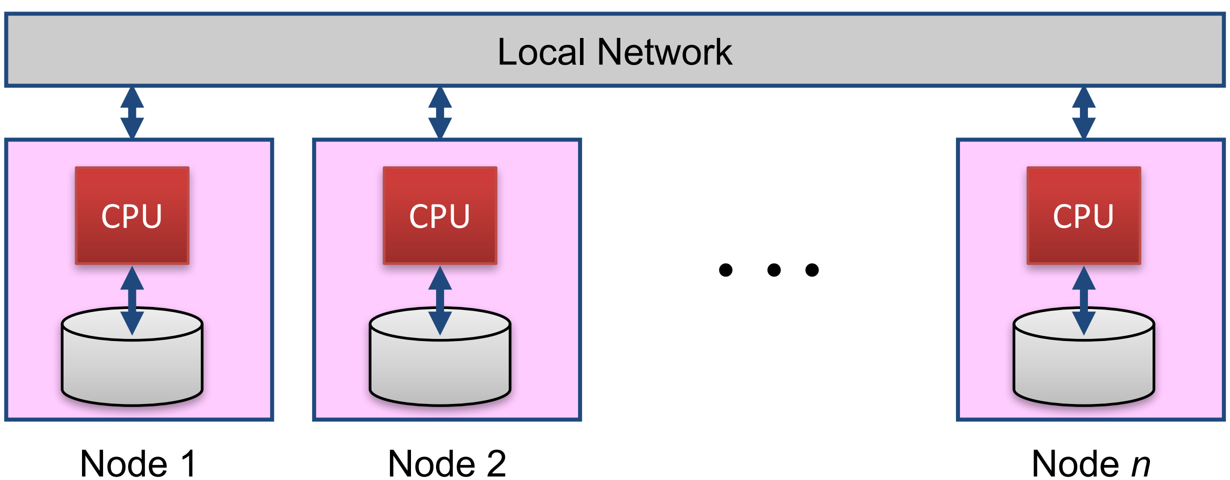
- An open source implementation of MapReduce framework
- Three components:
  - Hadoop Common Package (files needed to start Hadoop)
  - Hadoop Distributed File System: HDFS
  - MapReduce Engine
- HDFS requires data to be broken into blocks. Each block is stored on 2 or more data nodes on different racks.
- Name node: Manages the file system name space and keeps track of where each block is.



Hadoop (Cont.)

Distributed File System

– Replicate data in different place (typically 3 copies of each file)
  - If one node fails, data still available
– Logically, any node has access to any file
  - May need to fetch across network



Hadoop (Cont.)

MapReduce Programming Environment

- Data node: Constantly ask the job tracker if there is sth to do
- Job tracker: Assigns the map job to task tracker nodes that have the data or are close to the data (same rack)
- Task Tracker: Keep the work as close to the data as possible.

- Data nodes get the data if necessary, do the map function, and write the results to disks.
- Job tracker then assigns the reduce jobs to data nodes that have the map output or close to it.
- All data has a check attached to it to verify its integrity.

Word Count

Counting word for a document D

- If D can be fitted into memory, then?

```
Public void class WordCounter(String file){
    private Map word_map;

    public void count(String file){
        FileReader fr = new FileReader(file);
        BufferedReader br = new BufferedReader(fr);
        String line = “”;
        String[] words = null;
        word_map = new HashMap();

        while(!(line = br.readLine())){
            words = line.split();
            for(String word: words)
              word_map.put(word, word_map.get(word)+1)
        }
    }
}
```

Word Count (Cont.)

If D cannot be fitted into memory, then?

- Disk algorithm
- High Performance Computer
- Shared-memory cluster

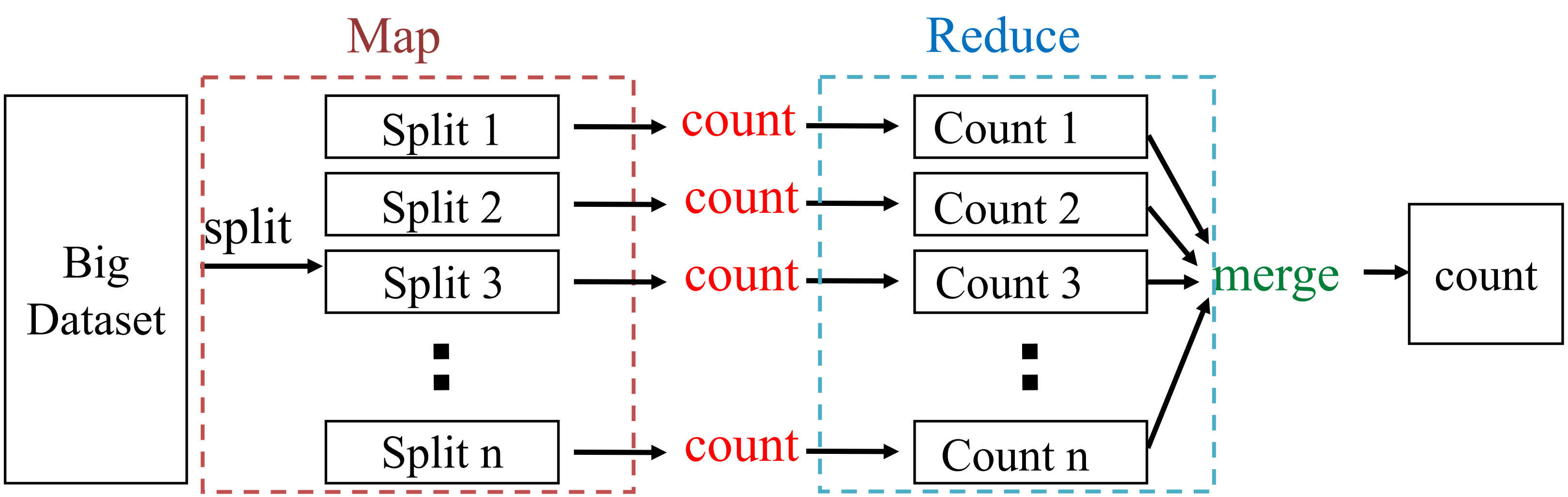
If you want to calculate the term frequency for entire Web pages as Google?

- 1,000,000,000*10KB/page = 10 TB
- Even larger amount of data such as 1PB, the scale that Google is facing

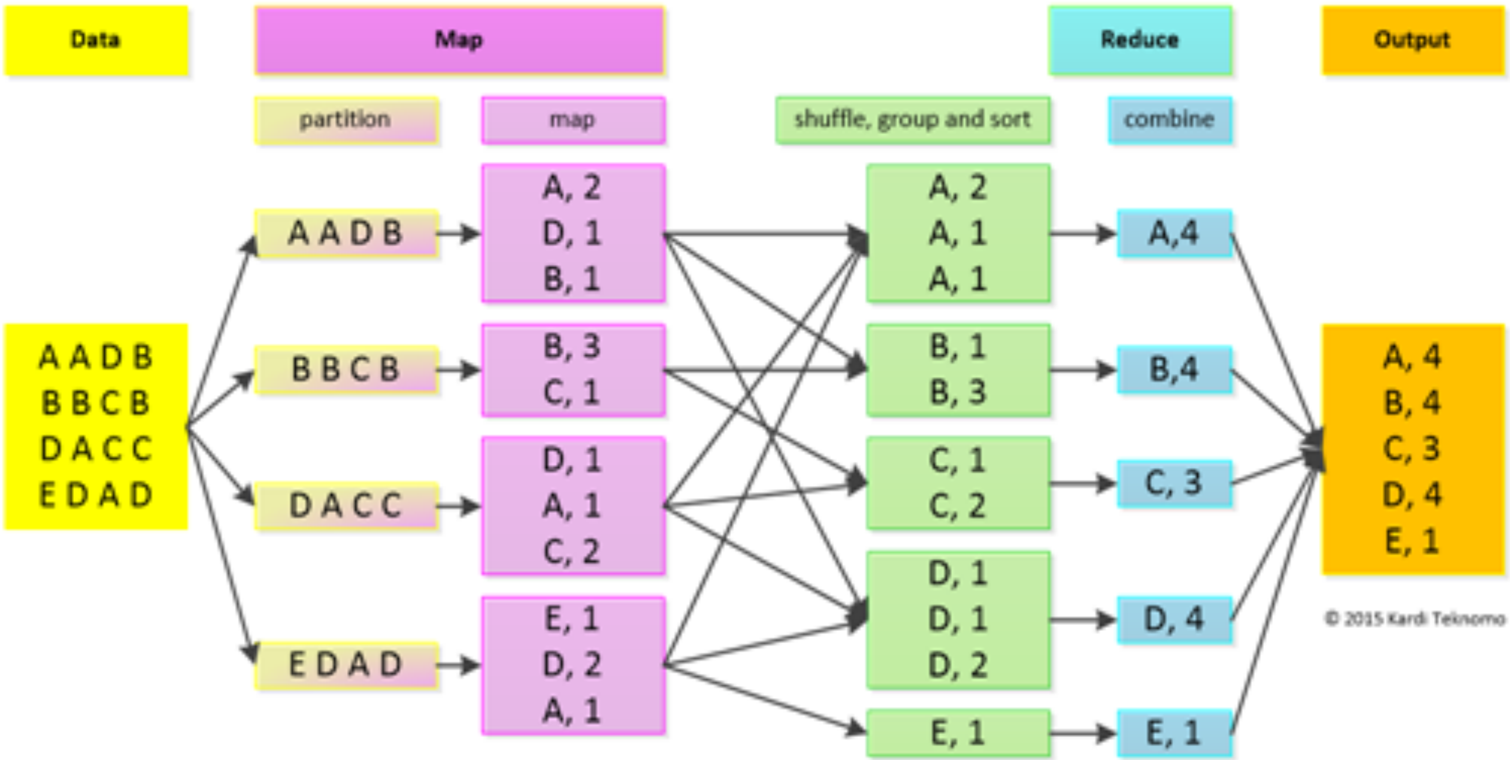
Word Count (Cont.)

```
map(string value)
   //key: document name
   //value: document contents
   for each word w in value
      EmitIntermediate(w, "1");
```

```
reduce(string key, iterator values)
   //key: word
   //values: list of counts
   int results = 0;
   for each v in values
      result += ParseInt(v);
   Emit(AsString(result));
```



How Does it Work?



Typical Tasks For Big Data Analytics

Analytics: Guide decision making by discovering patterns in data using statistics, programming, and operations research.

- SQL Analytics: Count, Mean, OLAP
- Descriptive Analytics: Analyzing historical data to explain past success or failures.
- Predictive Analytics: Forecasting using historical data.
- Prescriptive Analytics: Suggest decision options. Continually update these options with new data.
- Data Mining: Discovering patterns, trends, and relationships using Association rules, Clustering, Feature extraction
- Simulation: Discrete Event Simulation, Monte Carlo, Agent-based
- Optimization: Linear, non-Linear
- Machine Learning: An algorithm technique for learning from empirical data and then using those lessons to predict future outcomes of new data
- Web Analytics: Analytics of Web Accesses and Web users.

Ref: Michael Minelli, "Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today's Businesses," Wiley, 2013, ISBN:111814760X

Apache Hadoop Tools Stack

- Apache Hadoop: Open source Hadoop framework in Java. Consists of Hadoop Common Package (filesystem and OS abstractions), a MapReduce engine (MapReduce or YARN), and Hadoop Distributed File System (HDFS)
- Apache Mahout: Machine learning algorithms for collaborative filtering, clustering, and classification using Hadoop
- Apache Hive: Data warehouse infrastructure for Hadoop. Provides data summarization, query, and analysis using a SQLlike language called HiveQL. Stores data in an embedded Apache Derby database.
- Apache Pig: Platform for creating MapReduce programs using a high-level "Pig Latin" language. Makes MapReduce programming similar to SQL. Can be extended by user defined functions written in Java, Python, etc.

Ref: http://hadoop.apache.org/, http://mahout.apache.org, http://hive.apache.org/, http://pig.apache.org/

Apache Hadoop Tool Stack (Cont.)

- Apache Avro: Data serialization system. Avro IDL is the interface description language syntax for Avro.
- Apache HBase: Non-relational DBMS part of the Hadoop project. Designed for large quantities of sparse data (like BigTable). Provides a Java API for map reduce jobs to access the data. Used by Facebook.
- Apache ZooKeeper: Distributed configuration service, synchronization service, and naming registry for large distributed systems like Hadoop.
- Apache Cassandra: Distributed database management system. Highly scalable.

Ref: http://avro.apache.org/, http://cassandra.apache.org/, http://hbase.apache.org/, http://zookeeper.apache.org/

Apache Hadoop Tools Stack (Cont.)

- Apache Ambari: A web-based tool for provision, managing and monitoring Apache Hadoop cluster
- Apache Chukwa: A data collection system for managing large distributed systems
- Apache Sqoop: Tool for transferring bulk data between structured databases and Hadoop
- Apache Oozie: A workflow scheduler system to manage Apache Hadoop jobs

Ref: http://incubator.apache.org/chukwa/, http://oozie.apache.org/, https://sqoop.apache.org/, http://incubator.apache.org/ambari/

Apache Other Big Data Tools

- Apache Accumulo: Sorted distributed key/value store based on Google's BigTable design. 3rd Most popular NOSQL wide-column system. Provides cell-level security. Users can see only authorized keys and values. Originally funded by DoD.
- Apache Thrift: IDL to create services using many languages including C#, C++, Java, Python, Ruby, etc.
- Apache Beehive: Java application framework to allow development of Java based applications.
- Apache Derby: A RDBMS that can be embedded in Java programs. Needs only 2.6MB disk space. Supports JDBC (Java Database Connectivity) and SQL.

Ref: http://en.wikipedia.org/wiki/Apache_Accumulo, http://en.wikipedia.org/wiki/Apache_Thrift, http://en.wikipedia.org/wiki/Apache_Beehive, http://en.wikipedia.org/wiki/Apache_derby

Other Big Data Tools

- Cascading: Open Source software abstraction layer for Hadoop. Allows developers to create a .jar file that describes their data sources, analysis, and results without knowing MapReduce. Hadoop .jar file contains Cascading .jar files.
- Storm: Open source event processor and distributed computation framework alternative to MapReduce. Allows batch distributed processing of streaming data using a sequence of transformations.
- Elastic MapReduce (EMR): Automated provisioning of the Hadoop cluster, running, and terminating. Aka Hive.
- HyperTable: Hadoop compatible database system.

Ref: http://en.wikipedia.org/wiki/Cascading, http://en.wikipedia.org/wiki/Hypertable, http://en.wikipedia.org/wiki/Storm_%28event_processor%29

Other Big Data Tools (Cont.)

- Filesysem in User-space (FUSE): Users can create their own virtual file systems. Available for Linux, Android, OSX, etc.
- Cloudera Impala: Open source SQL query execution on HDFS and Apache HBase data
- MapR Hadoop: Enhanced versions of Apache Hadoop supported by MapR. Google, EMC, Amazon use MapR Hadoop.
- Big SQL: SQL interface to Hadoop (IBM)
- Hadapt: Analysis of massive data sets using SQL with Apache Hadoop.

Ref: http://en.wikipedia.org/wiki/CFilloeusydsetream_I_minp_aUlaserspace, http://en.wikipedia.org/wiki/Big_SQL, http://en.wikipedia.org/wiki/Cloudera_Impala, http://en.wikipedia.org/wiki/MapR, http://en.wikipedia.org/wiki/Hadapt

Summary

1. Big data has become possible due to low cost storage, high performance servers, high-speed networking, new analytics
2. Google File System, BigTable Database, and MapReduce framework sparked the development of Apache Hadoop.
3. Key components of Hadoop systems are HDFS, Avro data serialization system, MapReduce or YARN computation engine, Pig Latin high level programming language, Hive data warehouse, HBase database, and ZooKeeper for reliable distributed coordination.
4. Discovering patterns in data and using them is called Analytics. It can be descriptive, predictive, or prescriptive
5. Types of Databases: Relational, SQL, NoSQL, NewSQL, Key-Value Pair (KVP), Document, Columnar, Graph, and Spatial

References

- J. Hurwitz, et al., “Big Data for Dummies,” Wiley, 2013, ISBN:978-1-118-50422-2.
- Michael Minelli, “Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today‘s Businesses,” Wiley, 2013, ISBN:111814760X
- J. Dean and S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” OSDI 2004, http://research.google.com/archive/mapreduce-osdi04.pdf
- S. Ghemawat, et al., “The Google File System”, OSP 2003, http://research.google.com/archive/gfs.html
- F. Chang, et al., “Bigtable: A Distributed Storage System for Structured Data,” 2006, http://research.google.com/archive/bigtable.html
- A. Shieh, “Sharing the Data Center Network,” NSDI 2011, http://www.usenix.org/event/nsdi11/tech/full_papers/Shieh.pdf
- IBM. “What is MapReduce?,” http://www01.ibm.com/software/data/infosphere/hadoop/mapreduce/
- http://avro.apache.org/, http://cassandra.apache.org/
- http://hadoop.apache.org/, http://hbase.apache.org/
- http://hive.apache.org/, http://incubator.apache.org/ambari/
- http://incubator.apache.org/chukwa/, http://mahout.apache.org/
- http://oozie.apache.org/, http://pig.apache.org/
- http://zookeeper.apache.org/, https://sqoop.apache.org/

Quiz

- The 3V's that define Big Data are ________, ________, and _________.
- ACID stands for _________, _________, ________, and _________.
- BASE stands for _________, _________, ________, and ________ Consistency.
- ________ data is the data that has pre-set format.
- Data in ________ is the data that is streaming.

Solution to Quiz

- The 3V's that define Big Data are volume, velocity, and variety.
- ACID stands for Atomicity, Consistency, Isolation, and Durability.
- BASE stands for Basically Available, Soft, and Eventual Consistency.
- Structured data is the data that has pre-set format.
- Data in Motion is the data that is streaming.

3 Big Streaming Computation

Stream Processing, Apache Storm

Typical Paradigms for Data Analytics

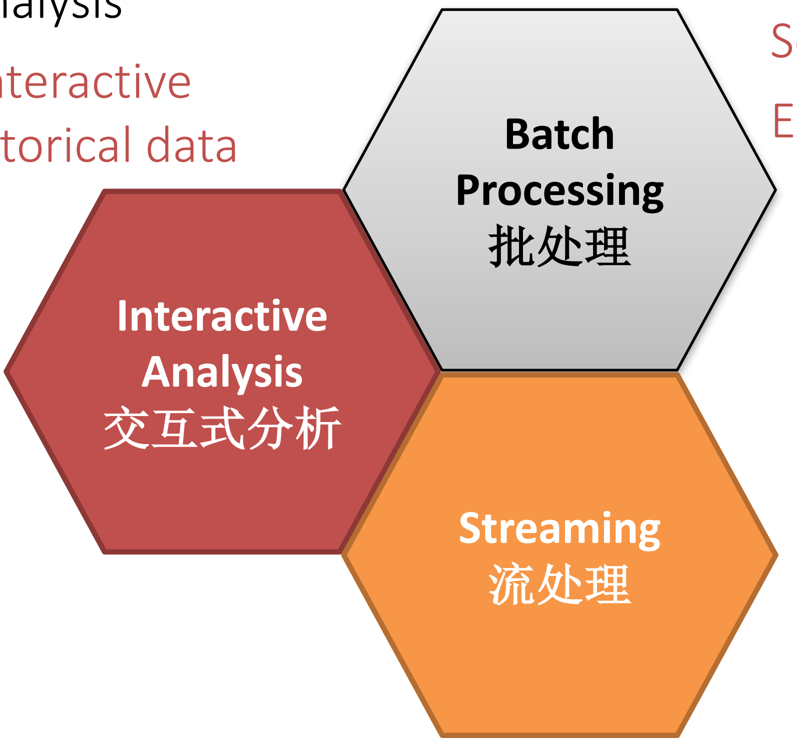
Exploratory analysis

Low latency interactive queries on historical data

Throughput

Sophisticated data processing

Enable “better” decisions



Real-time

Low latency queries on live data

Enable fast decisions

Big Streaming Data

Streaming Data

- Everything is in flux (万物皆流), Heraclitus
- Continuous and Non-deterministic
- Real-time processing (实时处理)
- Applications: Stock-trading management, Road traffic monitoring, Network fraud detection, Complex event processing, Click-stream analysis, etc.

Continuous Queries (CQs)

- Window-based query: tuple, time
- Sliding window: time >= '7:30' AND time < '8:00'
- Tumbling Window: every 5 minutes

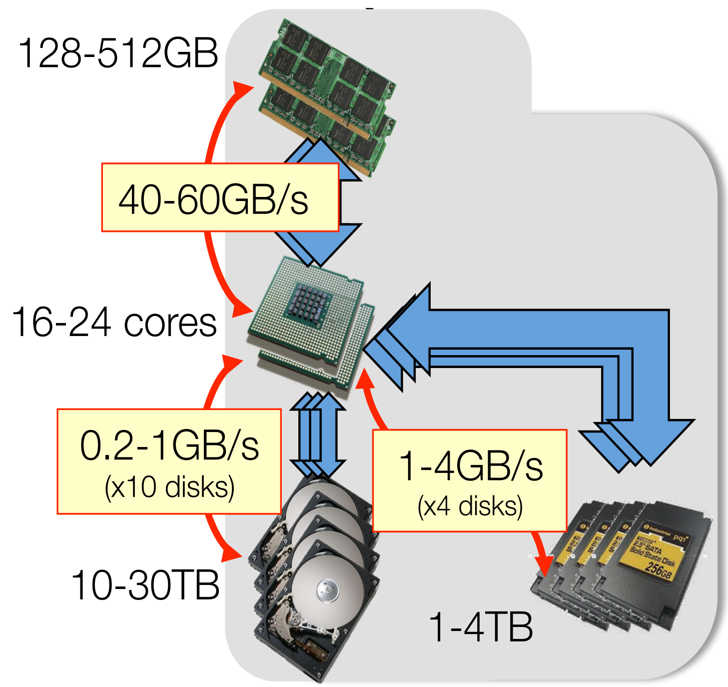
Example: Count the term occurrence of "Google" over a word stream "word_stream"

```
SELECT COUNT(num)
FROM word_stream [TIME 5 MINUTE ADVANCE 5 MINUTE]
WHERE tuple.key = Google
```

Realize Real-time Processing

Leverage Memory

The inputs of over 90% of jobs in Facebook, Yahoo!, and Bing clusters fit into memory



High-end datacenter node

The inputs of over 90% of jobs in Facebook, Yahoo!, and Bing clusters fit into memory

Realize Real-time Processing

Increase parallelism

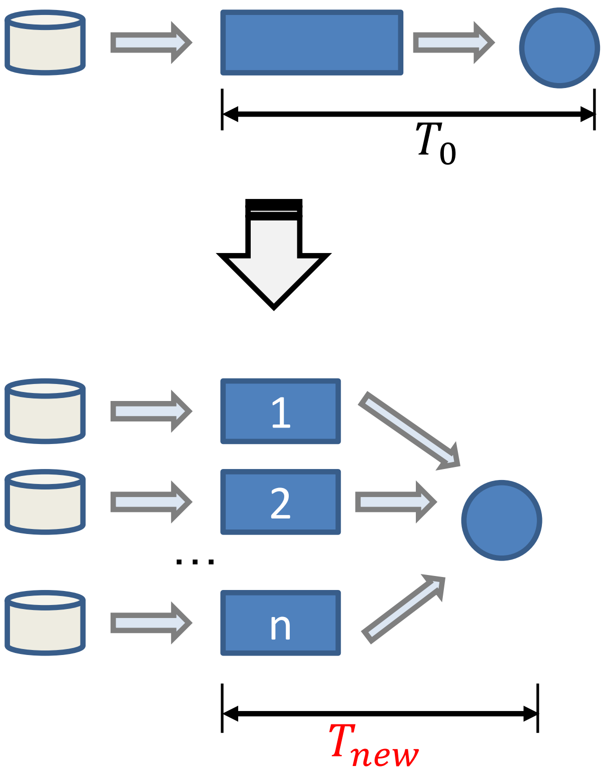
Reduce work per node improves latency

Techniques

Low latency scheduler

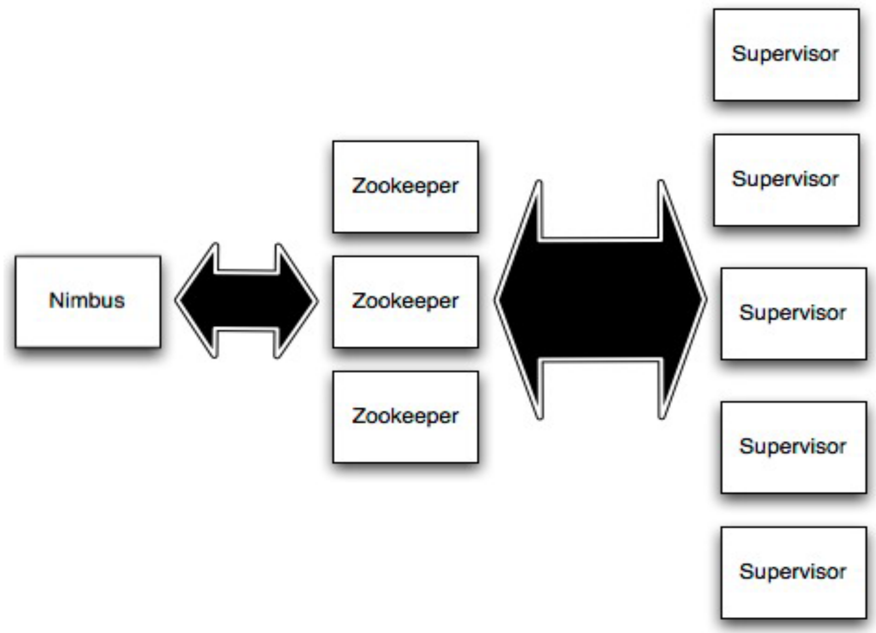
Efficient failure recovery

Optimization such as communication patterns: e.g., shuffle, broadcast



Streaming Processing With Apache Storm

Storm Cluster



Nimbus
Master node Similar to Hadoop JobTracker

Zookeeper
Used for cluster coordination
Preserve process state

Supervisor
Run worker processes

Key Concepts

- Topologies: Spouts, Bolts
- Streams, Stream groupings
- Tasks, Workers

Storm Components

Spout

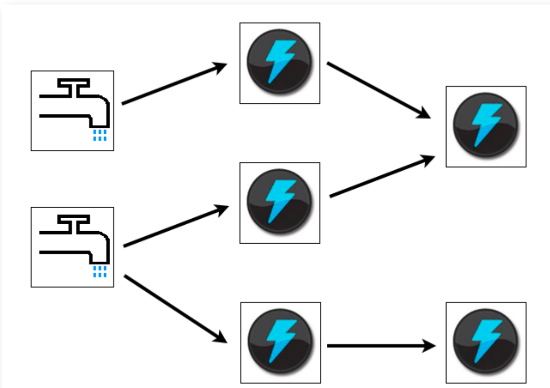
Ingest source streams
Kestrel queue, Kafka queue
Read from Twitter streaming API
HDFS, Hive
Database

Bolts

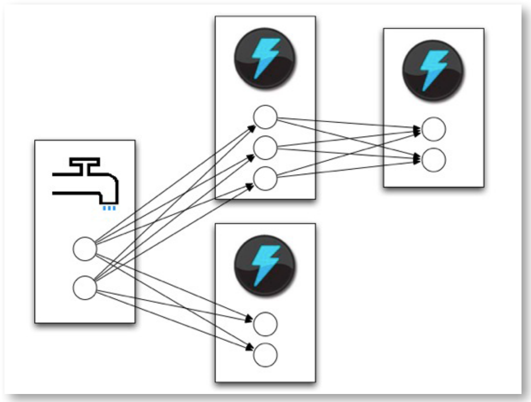
Processes Processes input streams and produces new streams
User defined functions,
Standard SQL operators: Filters, Aggregation, Joins

Topology, Tasks, and Task Execution

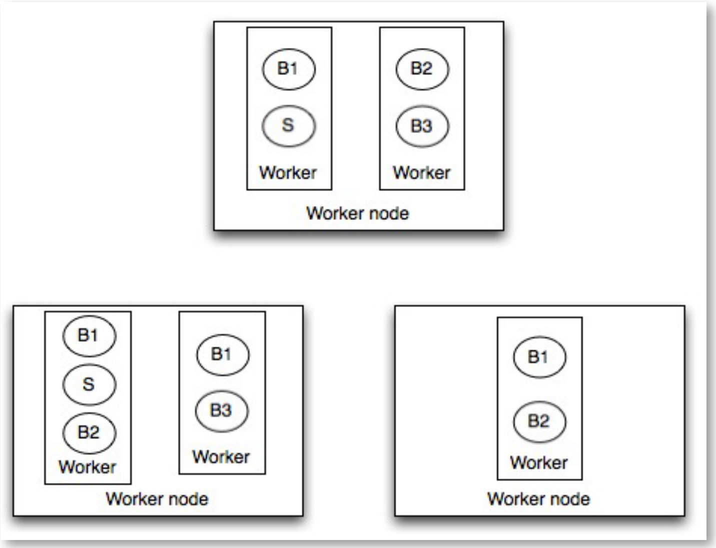
Network of spouts and bolts



Spouts and bolts execute as many tasks across the cluster

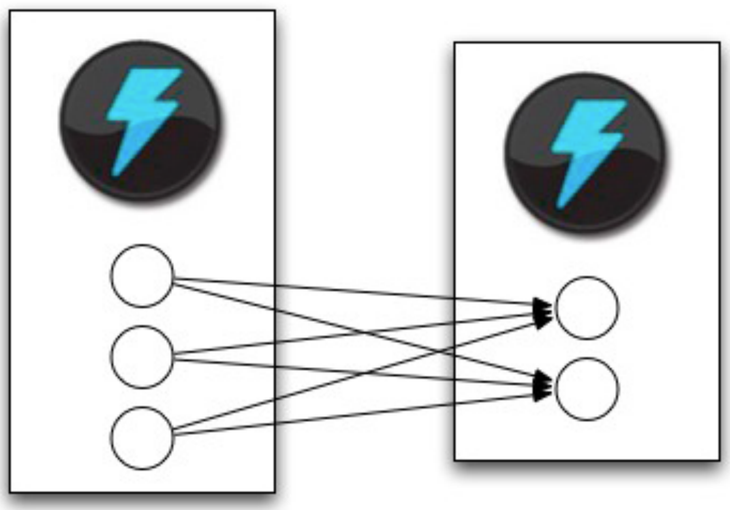


Tasks are spread across the cluster



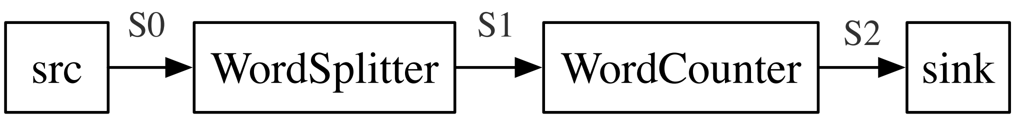
Stream Grouping

When a tuple is emitted, which task does it go to?



- Shuffle grouping: pick a random task
- Fields grouping: mod hashing on a subset of tuple fields
- All grouping: send to all tasks
- ...

Example: Streaming word count



Create a topology in storm

```
TopologyBuilder builder = new TopologyBuilder();
```

Define a spout in the topology with parallelism of 5 tasks

```
builder.setSpout("spout",
                 new KestrelSpout(
                     "kestrel.twitter.com"
                     22133,
                     "sentence_queue"),
                 5);
```

Create A Word Count Stream

Create a Bolt to split sentences into words with parallelism of 8 tasks

```
builder.setBolt("split", new SplitSentence(), 8)
        .shuffleGrouping("spout");
```

Create a bolt to receive word stream and to group it as key-value paire <word, count>

```
builder.setBolt("count", new WordCount(), 12)
        .fieldsGrouping("split", new Fields("word"));
```

Implementing Split Sentence

Implementing of SplitSentence bolt

```java
public static class SplitSentence extends ShellBolt implements IRichBolt {
    public SplitSentence() {
        super("python", "splitsentence.py");
    }

    public void declareOutputFields(OutputFieldsDeclarer declarer) {
        declarer.declare(new Fields("word"));
    }
}
```

```python
import storm

class SplitSentenceBolt(storm.BasicBolt):
    def process(self, tup):
        words = tup.values[0].split(" ")
        for word in words:
          storm.emit([word])
```

Implementing Word Count

```
public static class WordCount implements IBasicBolt {
    Map<String, Integer> counts = new HashMap<String, Integer>();

    public void prepare(Map conf, TopologyContext context) {
    }

    public void execute(Tuple tuple, BasicOutputCollector collector) {
        String word = tuple.getString(0);
        Integer count = counts.get(word);
        if(count==null) count = 0;
        count++;
        counts.put(word, count);
        collector.emit(new Values(word, count));
    }

    public void cleanup() {
    }

    public void declareOutputFields(OutputFieldsDeclarer declarer) {
        declarer.declare(new Fields("word", "count"));
    }
}
```

Word Count (Cont.)

Submitting topology to a cluster

```
Map conf = new HashMap();
conf.put(Config.TOPOLOGY_WORKERS, 10);

StormSubmitter.submitTopology("word-count", conf, builder.createTopology());
```

Running topology in local mode

```
LocalCluster cluster = new LocalCluster();

Map conf = new HashMap();
conf.put(Config.TOPOLOGY_DEBUG, true);

cluster.submitTopology("demo", conf, builder.createTopology());
```

4 Conclusion and Question

Conclusion

- The background of big data
  - Google File System, BigTable Database, and MapReduce framework sparked the development of Apache Hadoop.
- Big data fundamentals
  - GFS, MapReduce, BigTable
  - NoSQL, NewSQL
  - Hadoop and word count
  - Apache Big Data Analytical Tools
- Big streaming computation
  - Stream processing
  - Apache storm
  - Streaming word count

THANKS!

Q&A